Understanding node.js

Posted on 29/4/10 by Felix Geisendörfer

Node.js has generally caused two reactions in people I've introduced it to. Basically people either "got it" right away, or they ended up being very confused.

If you have been in the second group so far, here is my attempt to explain node:

- It is a command line tool. You download a tarball, compile and install the source.
- It let's you run JavaScript programs by typing 'node my_app.js' in your terminal.
- The JS is executed by the V8 javascript engine (the thing that makes Google Chrome so fast).
- Node provides a JavaScript API to access the network and file system

"But I can do everything I need in: ruby, python, php, java, ... !".

I hear you. And you are right! Node is no freaking unicorn that will come and do your work for you, sorry. It's just a tool, and it probably won't replace your regular tools completely, at least not for now.

"Get to the point!"

Alright, I will. Node is basically very good when you need to do several things at the same time. Have you ever written a piece of code and said "I wish this would run in parallel"? Well, in node everything runs in parallel, except your code.

"Huh?"

That's right, *everything runs in parallel, except your code*. To understand that, imagine your code is the king, and node is his army of servants.

The day starts by one servant waking up the king and asking him if he needs anything. The king gives the servant a list of tasks and goes back to sleep a little longer. The servant now distributes those tasks among his colleagues and they get to work.

Once a servant finishes a task, he lines up outside the kings quarter to report. The king lets one servant in at a time, and listens to things he reports. Sometimes the king will give the servant more tasks on the way out.

Life is good, for the king's servants carry out all of his tasks in parallel, but only report with one result at a time, so the king can focus. *

"That's fantastic, but could you quit the silly metaphor and speak geek to me?"

Sure. A simple node program may look like this:

```
var fs = require('fs')
  , sys = require('sys');
```

RSS Feeds

BLOG RSS
COMMENTS RSS

3261 readers BY FEEDBURNER

Recent Posts

- Releasing node-mysql 2.0.0-alpha
- How to write jQuery plugins
- Vim Workshop in Berlin (April 20)
- NPM - An intervention
- Testing node.js modules with Travis CI
- Private npm modules

Archive

443 Posts, **4608 Comments**

We are on air since 2006. Take a tour through the blog archive.

Search

Recent Comments

William: thanks, fixed ..
Hi Felix, The Upgra ..
Maybe you can write so ..
This is very nice plug ..
Nice article! Somet ..
That's fine ognian, th ..
Really nice work! I wa ..
@Jörn: Sorry, thought ..
Thanks didn't quite un ..
Well you could expose ..

```
fs.readFile('treasure-chamber-report.txt', function(report) {
  sys.puts("oh, look at all my money: "+report);
});

fs.writeFile('letter-to-princess.txt', '...', function() {
  sys.puts("can't wait to hear back from her!");
});
```

Your code gives node the two tasks to read and write a file, and then goes to sleep. Once node has completed a task, the callback for it is fired. But there can only be one callback firing at the same time. Until that callback has finished executing, all other callbacks have to wait in line. In addition to that, there is no guarantee on the order in which the callbacks will fire.

"So I don't have to worry about code accessing the same data structures at the same time?"

You got it! That's the entire beauty of JavaScripts single-threaded / event loop design!

"Very nice, but why should I use it?"

One reason is efficiency. In a web application, your main response time cost is usually the sum of time it takes to execute all your database queries. With node, you can execute all your queries at once, reducing the response time to the duration it takes to execute the slowest query.

Another reason is JavaScript. You can use node to share code between the browser and your backend. JavaScript is also on its way to become a really universal language. No matter if you did python, ruby, java, php, ... in the past, you've probably picked up some JS along the way, right?

And the last reason is raw speed. V8 is constantly pushing the boundaries in being one of the fastest dynamic language interpreters on the planet. I can't think of any other language that is being pushed for speed as aggressively as JavaScript is right now. In addition to that, node's I/O facilities are really light weight, bringing you as close to fully utilizing your system's full I/O capacities as possible.

"So you are saying I should write all my apps in node from now on?"

Yes and no. Once you start to swing the node hammer, everything is obviously going to start looking like a nail. But if you're working on something with a deadline, you might want to base your decision on:

- Are low response times / high concurrency important? Node is really good at that.
- How big is the project? Small projects should be fine. Big projects should evaluate carefully (available libraries, resources to fix a bug or two upstream, etc.).

"Does node run on Windows?"

No. If you are on windows, you need to run a virtual machine (I recommend VirtualBox) with Linux. Windows support for node is planned, but don't hold your breath for the next few months unless you want to help with the port.

"Can I access the DOM in node?"

Excellent question! No, the DOM is a browser thingy, and node's JS engine (V8) is thankfully totally separate from all that mess. However, there are people working on implementing the DOM as a node module, which may open very exciting possibilities such as unit testing client-side code.

"Isn't event driven programming really hard?"

That depends on you. If you already learned how to juggle AJAX calls and user events in the browser, getting used to node shouldn't be a problem.

Keep an eye on

- Transloadit
- tus - resumable file upload protocol
- Kevin van Zonneveld
- Chris Hartjes
- Mark Story
- Webbosaurus

Either way, test driven development can really help you to come up with maintainable designs.

"Who is using it?"

There is a small / incomplete list in the node wiki (scroll to "Companies using Node"). Yahoo is experimenting with node for YUI, Plurk is using it for massive comet and Paul Bakaus (of jQuery UI fame) is building a mind-blowing game engine that has some node in the backend. Joyent has hired Ryan Dahl (the creator of node) and heavily sponsors the development.

Oh, and Heroku just announced (experimental) hosting support for node.js as well.

"Where can I learn more?"

Tim Caswell is running the excellent How To Node blog. Follow #nodejs on twitter. Subscribe to the mailing list. And come and hang out in the IRC channel, *#node.js* (yes, the dot is in the name). We're close to hitting the 200 lurker-mark there soon :).

I'll also continue to write articles here on debuggable.com.

That's it for now. Feel free to comment if you have more questions!

--fg

*: The metaphor is obviously a simplification, but if it's hard to find a counterpart for the concept of non-blocking in reality.

Did you like this blog post? If so, please consider subscribing to the Blog RSS feed.

(Theoretically) Related Posts

- Understanding hidden classes in v8
- node.js - Dealing with uncaught exceptions
- Streaming UTF-8 (with node.js)
- node.js
- Parsing form data with node.js
- Unit testing with node.js
- Testing node.js modules with Travis CI
- Streaming file uploads with node.js
- Parsing file uploads at 500 mb/s with node.js
- Releasing node-mysql 2.0.0-alpha

You can skip to the end and add a comment.

26 COMMENTS | ADD COMMENT

Mark Story said on Apr 29, 2010:

Great introduction Felix, I've been reading the Node.js docs and it looks pretty amazing. Just need to get the time to get my hands dirty :)

Felix Vargas said on Apr 29, 2010:

Hey! thanks for the intro...

Karl G said on Apr 29, 2010:

Hi Felix. Good job on the node evangelism. I just have one complaint.

> V8 is constantly pushing the boundaries in being the fastest dynamic language interpreter on the planet. I can't think of any other language that is being pushed for speed as aggressively as JavaScript is right now.

This is, unfortunately, not true. To my knowledge, the fastest dynamic language interpreter is Mike Pall's LuaJIT. There was an extended exchange between Mike, Brendan Eich, and Andreas Gal on Lambda the Ultimate at the end of March [1] but in brief, js language semantics prevent some optimizations that LuaJIT has in place.

[1] http://lambda-the-ultimate.org/node/3851

Botanicus said on Apr 29, 2010:

Hi, I like this blog, thanks for it! Anyway I'm writing about Node.js as well at http://blog.101ideas.cz

---

Felix Geisendörfer said on Apr 30, 2010:

Karl G: I agree with you partially, so I changed the wording to 'one of the fastest'.

However, I don't see evidence that LuaJIT is the *fastest* interpreter. At least we'd have to decide on an exact definition of "fast" first :).

LuaJIT seems to do very well in the game of language benchmarks, but those seem rather un-realistic at times. I mean I would consider anything related to closures much more important (speed-wise), than any raw number crunching.

Either way, it's great to see the kind of innovation going on there.

Mislav said on May 01, 2010:

I loved the metaphor.

Vipul Limbachiya said on May 05, 2010:

Great article and introduction to node.js.

It helped me to understand very basic thing about it and where and why to use it.

Thanks for sharing

Vipul

Paul Grayson said on Jul 20, 2010:

Nice introduction Felix. A lot of people seem to have heard of node.js but miss the evented/non-blocking aspect and think it's just another server stack - all be it a JS one. I'll be pointing them at this article in future ;-)

Stuart said on Jul 26, 2010:

Love your RSS feed bug! Gave me a laugh.

---

bruce said on Jul 29, 2010:

check out simon willison's blog.

http://simonwillison.net/2009/Nov/23/node/

he uses an analogy with bunnies :)

http://www.slideshare.net/simon/evented-io-based-web-servers-explained-using-bunnies

loving your work with multipart uploads using node.

---

asksuperuser said on Aug 14, 2010:

why wouldn't it run on windows since v8 can ?

Felix Geisendörfer said on Aug 14, 2010:

asksuperuser: It runs on cygwin now. Full windows support should be possible as well, there just needs to be somebody to actually get a build going.

Jonah Bron said on Aug 18, 2010:

Cool, I didn't really understand what node.js was before. Nice introduction, thanks!

Alexis Lee said on Aug 19, 2010:

Thanks, this is just what I needed.

For nonblocking, I may be misunderstanding of course... I imagine a servant standing in front of the king, making a phone call.

Of course the king can hire a peon to wait for the servant to finish his call, tell the peon how to handle the results and ask it to report back to him - but that's a lot of trivia to deal with a rude servant. Off with his head!

Alex C said on Aug 20, 2010:

Wait.

I'm still not clear... is this something I would use for a web page or for a "stand alone / compiled app" ?
The bit about asynchronous database calls made me just a little bit giddy. Sounds really cool.

I worry that I'm not quite smart enough to leverage the power of that but maybe we're all a little scared (and excited) around new languages... but I guess you don't know until you try.

Alex C said on Aug 20, 2010:

like is it just 'automatically faster' or do I have to change my program architecture to take advantage?

Sam said on Aug 25, 2010:

Great explanation of the most important web programming language. What are the other alternatives?

Geries said on Sep 06, 2010:

Thanks for the explanation, was kind of lost.

Here is a list of modules for node.js http://github.com/ry/node/wiki/modules#database In case anybody is wondering (like I did) what DB to use. Also, you will find parsers, template systems, middleware, frameworks, etc. http://github.com/ry/node/wiki/modules

---

smegroume said on Sep 12, 2010:

nice blog man also really like the design

---

www.tech-wd.com said on Sep 13, 2010:

Of course the king can hire a peon to wait for the servant to finish his call, tell the peon how to handle the results and ask it to report back to him - but that's a lot of trivia to deal with a rude servant. Off with his head!

---

a-dizzle said on Sep 30, 2010:

Apparently I am taking a calculus class from Ryan Dahl's mother. She was talking about how her son made a bunch of money making this thing called "node.js" the other day. Small world...

---

wjr said on Sep 30, 2010:

How can I use this if my hosting site is something like godaddy.com or dreamhost?

---

bruce said on Oct 01, 2010:

just checking google 'nodejs hosting' came us these articles...

http://remysharp.com/2010/02/14/slicehost-nodejs-websockets/

http://blog.nodejitsu.com/nodejs-cloud-server-in-three-minutes

this is a list from a stack overflow thread... hope that helps.
also joyent seemed to pop up alot.

WebFaction
Heroku – Closed Beta

ElusiveHippo – Not Working

Prgmr

Slicehost

Linode

Amazon EC2

Rackspace Cloud

VPS.Net

Joyent

no.de – Closed Beta

DreamHostPS – Need a total reconfiguration of virtual machines

Webbynode – setup guide here

Cliff James said on Oct 20, 2010:
Just for clarification. Node isn't a generic tool for doing event-based multi-threading right? Isn't more about event-based async IO. I mean if you were developing some kind of processor intensive task (no IO) that you wanted split across multiple cores, I don't see how Node helps you.

Felix Geisendörfer said on Oct 20, 2010:

Cliff James: It depends, but generally the distribution of CPU bound algorithms is not a strength of node, because you have to split tasks up into processes and use sockets for communicating between them (which is significant overhead in certain scenarios). So look at your problem and see if you can life with the IPC overhead or not.

Marko said on Oct 22, 2010:

very good introduction, best I have found for #nodejs...thanx

---

This post is too old. We do not allow comments here anymore in order to fight spam. If you have real feedback or questions for the post, please contact us.